05

Usando eclipse

Transcripción

[00:00] Entonces ahora ya tenemos nuestra clase ejemplo lista para agregarle nuestro método "Hola Mundo". Vamos a seguir los mismos pasos que seguimos para crear nuestro "Hola Mundo" en el archivo de texto normal. Pero antes de eso quisiera acomodar un poco mi entorno de desenvolvimiento.

[00:20] En este caso ya mencionamos las vistas, mencionamos las perspectivas y yo sé que aún no ha quedado del todo claro absolutamente para qué son y para qué sirven. No se preocupen, yo creo que con la práctica ya con el uso del IDE vamos a ir descubriendo mejor cuál es la real finalidad de tener tantas ventanitas abiertas aquí.

[00:44] Entonces yo voy a cerrar esto, voy a cerrar esto de aquí y voy a trabajar con este entorno bien básico. Entonces, vamos a comenzar a escribir nuestro método. En este caso ya comenzamos escribiendo public, que es nuestro modificador de acceso, y si yo no recuerdo bien por ejemplo cuál era la sintaxis de aquel método extraño que escribí en el bloc de notas, voy a comenzar solo con void.

[01:17] Y yo recuerdo que se llamaba Main. Perfecto. Aquí abro las llaves y primera cosa que yo puedo dotar es que el IDE automáticamente ya me completa el cierre de llaves, es una ayuda. Ahora yo recuerdo que escribía system alguna cosa para poder imprimir aquella línea de "Hola Mundo" ¿recuerdan?

[01:45] Entonces comenzamos system.out, perfecto, lo encontré aquí. Si se dan cuenta, el IDE me va ayudando con las opciones que necesito, él trata de adivinar más o menos qué es lo que estoy buscando y me ayuda a autocompletar. Si le doy enter, vemos que automáticamente System cambió a mayúscula. Yo había escrito con minúscula. ¿Por qué?

[02:12] Porque la referencia bien escrita es System en mayúscula. Entonces, si yo estoy equivocado, ya el IDE automáticamente dice: "Ah, no, este chico está buscando la librería System, voy a ayudarlo, voy a corregir automáticamente." Si yo doy un punto aquí, tengo los métodos que están dentro de out al mismo tiempo.

[02:34] Si yo diera "Ctrl + espacio" dentro de System, yo vería los métodos de System. Conforme yo voy avanzando el IDE me va ayudando mostrándome cada método que está dentro de la clase que yo estoy explorando, y al mismo tiempo me ayuda con pequeña documentación esta de aquí.

[02:54] Él me dice más o menos qué es lo que hace ese método, para qué sirve, qué argumentos acepta y qué es lo que él me retorna. Entonces yo recuerdo que el método era print y tenemos varios tipos de print. Yo tengo un println, de print line, que en este caso es un print line de string. Entonces voy a seleccionarlo. Perfecto.

[03:17] Y aquí me está pidiendo automáticamente, él ya me dio el punto y coma al final y ya me direccionó diciéndome: "Aquí en esta X, aquí vas a poner un string". Y yo voy a poner mi string, que en este caso es: "Hola Mundo". Perfecto. ¿Sería este método el que yo necesito para ejecutar "Hola Mundo" en la consola? Yo creo que aquí le falta algo, pero vamos a probar, vamos a intentarlo.

[03:50] Vamos a darle run, vamos a decirle que corra ejemplo, que lo guarde antes de correr, le damos okay y nos dice: "No se ha encontrado el método principal en la clase Ejemplo. Defina el método principal". Y nos da esta

sugerencia: public static void main string args. En este caso este es el nombre correcto que yo estaba necesitando poner aquí.

[04:19] Vemos cómo el IDE automáticamente abrió una consola, una vista de consola, que es la mima terminal que nosotros ya hemos venido trabajando antes, solo que ya contenida adentro del IDE. Esto nos evita la necesidad de tener que abrir otra ventana u otro programa para poder ejecutar el código.

[04:40] Entonces, si yo le doy aquí public static void main string, argentinos, vamos a ver qué dice aquí, vamos a volver a ejecutar esto, le damos que guarde la clase Ejemplo, y ahora sí tenemos un "Hola Mundo". ¿Será que esto solo funciona dentro del IDE?

[05:17] Vamos a probar si accediendo desde una consola externa vamos a tener el mismo resultado. Para eso abrimos nuestra consola y entramos a nuestro workspace. Para eso entramos a Eclipse workspace, vamos a un dir, y vemos que aquí tenemos algunos metadatos propios de workspace y nuestro proyecto ya va primeros pasos en una carpeta.

[05:49] Entonces, entramos ahí, damos otro dir para ver qué tenemos ahí, y vemos que tenemos un archivo punto classpath, archivo project, settings, bin y SRC. Pero yo aquí solamente estoy viendo esto de acá y SRC. Entonces, ¿qué son estos otros archivos que yo estoy viendo aquí y por qué no los veo aquí? Primer punto.

[06:15] Estamos en una vista de Package Explorer. Entonces, esta vista solamente nos va a permitir ver los archivos con los cuales nosotros estamos teniendo interacción. En el caso de los archivos classpath, project, settings, bin, son archivos con los cuales nosotros no tenemos interacción, no vamos a editar directamente.

[06:37] Son archivos generados, ya sea archivos resultados de compilación de código Java, que están dentro de la carpeta bin, o archivos generados por el propio IDE, como classpath, project y settings. ¿Pero qué pasaría si yo deseo

igual de cualquier forma verlos aquí dentro de Package Explorer? Entonces vamos aquí a window y tenemos aquí dos cosas muy importantes que son show view y perspective.

[07:09] Show view nos va a permitir digamos consolas extras, un log de errors, Javadoc, navigator, que es lo que nosotros vamos a usar en algún momento, pero básicamente nos va a permitir abrir pequeñas ventanitas extras, pequeñas vistas, para que nos dé un poco más de soporte en nuestro proceso de desenvolvimiento.

[07:34] Entonces vamos a abrir navigator, va a abrir aquí una nueva vista, perfecto. Y aquí navigator sí tenemos la vista total de todos los archivos que están dentro de Java primeros pasos: tenemos project, classpath, settings y bin. Y aquí dentro es donde está nuestro archivo punto class. ¿Por qué el IDE hace esta separación? Porque nosotros nunca vamos a editar directamente un archivo punto class.

[08:02] Porque nosotros nunca vamos a editar directamente un archivo punto class. Este archivo tiene que ser generado en base al archivo punto Java, que es lo que nosotros estamos viendo aquí. Ese es uno de los usos más comunes de las vistas, ir poco a poco personalizando nuestro entorno de desenvolvimiento.

[08:24] Ahora, un conjunto de vistas hace lo que sería la perspectiva. Si vamos a perspectivas, tenemos algunas que son más apropiadas para debug, que nos van a ofrecer digamos vistas más puntuales para ese fin que tenemos, o nosotros podemos crear nuestro propio conjunto de vistas y salvar nuestra perspectiva personalizada.

[08:52] Creo que ahora ya quedó un poco más claro para qué es que sirve esto. Entonces, volvemos a la terminal para probar. Y ya que estamos aquí en Java primeros pasos y sabemos que nuestro ByteCode está aquí en bin, si nosotros entramos a SRC y queremos ejecutar Java ejemplo, nos va a dar un error, porque aquí no está el archivo punto class.

[09:19] Tenemos que ir donde está el archivo punto class. Cd bin, damos nuevamente Java ejemplo y vemos el "Hola Mundo". Es así como ya entendemos por qué es que el IDE facilita mucho la vida de un programador. Por ejemplo si yo tiro aquí el punto y coma final, automáticamente él me va a marcar un error. Él me va a decir: "Inserta un punto y coma para completar esta línea de código". Perfecto.

[09:52] Si yo hiciera esto de aquí, también me daría un error aquí. ¿Por qué? Porque me estaría diciendo: "Necesitas declarar aquí el ID de la variable, nos está pidiendo el tipo de dato. Si bien no es la descripción de error más detallada, ya nos ayuda mucho a la hora de entender por qué no compila ese código.

[10:19] Espero les haya quedado un poco más claro cuál es la utilidad del IDE. Ahora que tenemos nuestro ambiente ya preparado, vamos a comenzar a escribir código Java, a probar métodos, funciones y algunas cosas un poco más avanzadas.

[10:36] Recordemos ahora un pequeño review de lo que hemos visto. Sabemos la Virtual Machine, sabemos cómo generar ByteCode, sabemos dónde se está guardando ese ByteCode, sabemos que podemos escribir código en un simple bloc de notas y ahora sabemos que tienes herramientas que nos ayudan a escribir ese código mucho más fácilmente.

[10:55] Nos vemos en el siguiente video y espero que estén tan emocionados como yo de ya ir avanzando, ir aprendiendo cada vez más en este maravilloso mundo de Java. Chau, chau.